Aula 5
Movimento em 2D

Aula 5 - Objetivos

Ao final desta aula você deve estar apto a:

1) Usar o método de Euler para "integrar" a equação de movimento em 2D.
2) Fazer gráficos de *trajetórias* em 2D, ponto a ponto.
3) Usar as estruturas **if** **(...)** **else** **(...)** **end** e **while** **(...)** **end** do MatLab.
4) Usar *interpolação linear* para estimar o alcance de um lançamento.

Tarefas:

1. Tarefa 1 (Partes 1 e 2): Escrever um script para calcular e plotar a trajetória (x(t),y(t)) de um corpo lançado com um ângulo θ em relação à horizontal.
   Tempo aproximado: 30 min.

2. Tarefa 2: Adaptar o script da Tarefa 1 e usar interpolação linear para calcular o alcance $x_A(\theta)$ e armazenar em um vertor para θ, fazendo um gráfico $x_A(\theta)$ vs θ.
   Tempo aproximado: 20 min.

Cada tarefa deve ser entregue em um arquivo separado

Movimento em 2D: força constante

Força **constante** na direção y: $$\vec{F}_R = -mg\,\mathbf{j}$$

2a Lei de Newton

$$\boxed{\left\{\begin{array}{rcl} m\dfrac{dv_x}{dt} & = & F_x = 0 \\ \\ m\dfrac{dv_y}{dt} & = & F_y = -mg \end{array}\right.}$$

Posição (x,y)

$$\boxed{\left\{\begin{array}{rcl} \dfrac{dx}{dt} & = & v_x \\ \\ \dfrac{dy}{dt} & = & v_y \end{array}\right.}$$

Temos um **sistema de equações diferenciais** (acopladas)

Movimento em 2D: lançamento

Método de Euler:
$$\frac{dv_x}{dt} \approx \frac{v_x(t+\Delta t) - v_x(t)}{\Delta t}$$
(o mesmo para y):
$$\frac{dx}{dt} \approx \frac{x(t+\Delta t) - x(t)}{\Delta t}$$

Ou seja, **dados** x(t) y(t) v_x(t) e v_y(t) podemos calcula-los em t+ Δt:

$$\boxed{\left\{\begin{array}{rcl} x(t+\Delta t) & = & x(t) + v_x(t)\ \Delta t \\ y(t+\Delta t) & = & y(t) + v_y(t)\ \Delta t \\ v_x(t+\Delta t) & = & v_x(t) \\ v_y(t+\Delta t) & = & v_y(t) - g\ \Delta t \end{array}\right.}$$

Solução numérica: método de Euler

Discretização!

$$t_n = (n-1)\,\Delta t$$

$$v_x(t) \approx v_x(t_n) \equiv (v_x)_n \qquad x(t) \approx x(t_n) \equiv x_n$$

$$v_y(t) \approx v_y(t_n) \equiv (v_y)_n \qquad y(t) \approx y(t_n) \equiv y_n$$

Posição e velocidade são dados em intervalos *discretos*!

Fórmula recursiva:

$$\begin{cases} x_{n+1} & = & x_n + (v_x)_n \,\Delta t \\ y_{n+1} & = & y_n + (v_y)_n \,\Delta t \\ (v_x)_{n+1} & = & (v_x)_n \\ (v_y)_{n+1} & = & (v_y)_n - g\,\Delta t \end{cases}$$

- Começando em $t_1=0$ [dados $x(0), v_x(0), y(0), v_y(0)$] podemos calcular tudo em t_2 [$x_2, (v_x)_2, y_2\ (v_y)_2$].
- Temos tudo em t_2, calculamos tudo em t_3.... e assim por diante!

Aula 5 – Tarefa 1 – Parte 1 (~15 min)

Um corpo é lançado a partir do solo com velocidade inicial v=10m/s a um ângulo θ=45º com a horizontal.

- *Calcule sua posição (x,y) nos tempos $t_n=(n-1).\Delta t$ de $t_1=0$ até $t_N=3s$ com passo de* ***Δt=0.1s*** *usando o método de Euler.*
- *Para cada passo, imprima t_n ,$x(t_n)$ e $y(t_n)$ com 5 casas e imprima* ***o ponto*** *(x,y) em um gráfico.* (Dica 1)

- ***Discuta:*** *o que ocorre para tempos longos? Faz sentido físico? Você confia na sua simulação??*

Dica 1 : Para imprimir um ponto (x,y) [**se x e y são números(*)**] use:

plot(x,y,'b-o').

(*) ou seja, vetores de um único elemento.

Estruturas condicionais: “if” e “while”

```
%% Exemplo de if e while
clear;
ii=0; % Inicializa
while (ii<5)
    fprintf('O valor de ii é %i \n',ii);
    if (ii==3) % Cuidado com o == !
        disp('Ok,  ii é 3 !');
    else
        disp('Ops, ii nao é 3');
    end
    % end if
    ii=ii+1; % Importante!!!
end
% end while
```

executa ENQUANTO ii for menor que 5

executa SE ii for igual 3

executa SE ii nao for igual a 3

Atualiza ii ou você entra em loop infinito!

Aula 5 – Tarefa 1 – Parte 2 (~15 min)

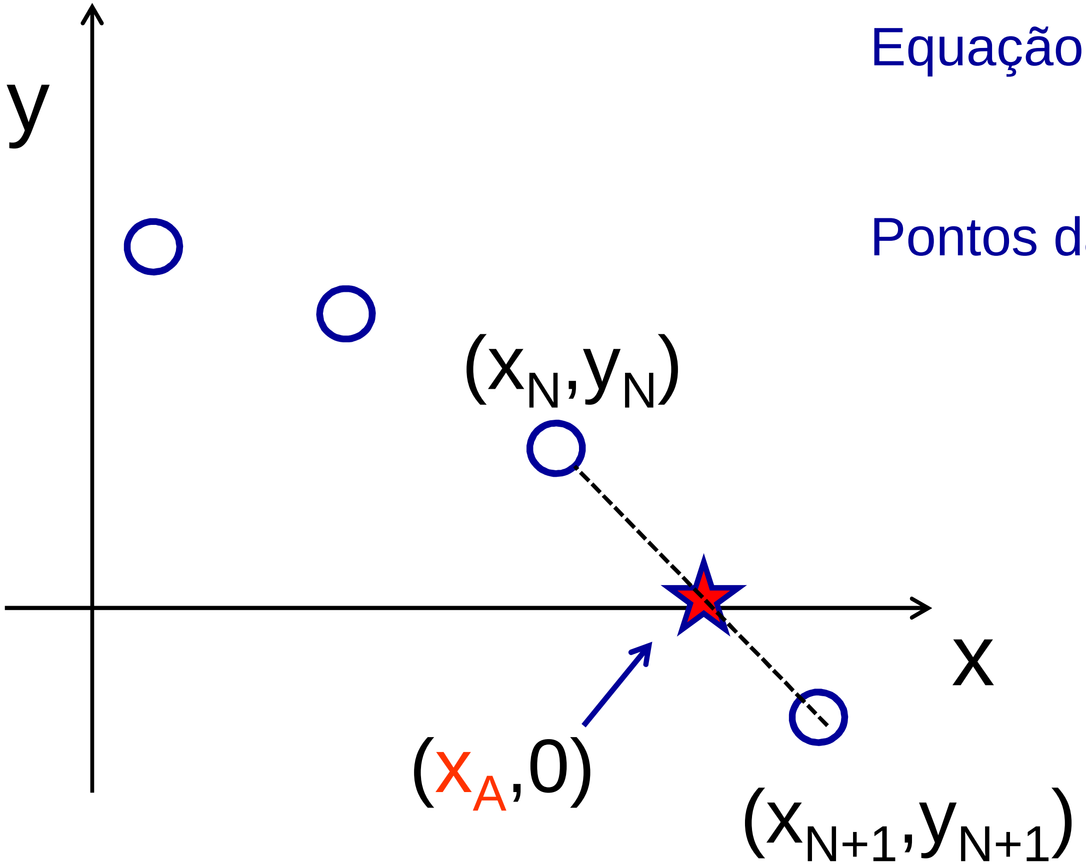
Um corpo é lançado a partir do solo com velocidade inicial v=10m/s a um ângulo θ=45º com a horizontal.

- *Calcule sua posição (x,y) nos tempos $t_n=(n-1).\Delta t$ a partir de $t_1=0$ com passo de* ***Δt=0.1s*** *usando o método de Euler.*
- *Para cada passo, imprima t_n ,$x(t_n)$ e $y(t_n)$ com 5 casas decimais e imprima* ***o ponto*** *(x,y) em um gráfico.* Dica 1: use **plot(x,y,'b-o')**
- *Pare o procedimento quando y for <0 (queda ao chão)* (Dica 2)
- *Estime o valor máximo de x (alcance $\boldsymbol{x_A}$). Como obter uma melhor aproximação para este valor?* (Dica 3)

Dica 2 : Use um loop while com a condição (y>=0) na atualização de x,y!

Dica 3 : Armazene os dois últimos valores de x e y e use uma interpolação linear para estimar $\mathbf{x_A}$ sabendo que $\mathbf{y_A=0}$.

Interpolação Linear



Equação da reta: $\boxed{y = Ax + B}$

Pontos da reta: $\begin{cases} y_N = Ax_N + B \\ y_{N+1} = Ax_{N+1} + B \end{cases}$

Resolvendo para A e B:

$$\begin{cases} A = \dfrac{y_{N+1} - y_N}{x_{N+1} - x_N} \\ B = y_N - Ax_N \end{cases}$$

Estimativa para x_A! $0 = Ax_A + B \Rightarrow x_A = \dfrac{-B}{A}$

Aula 5 – Tarefa 2 (~20 min)

Um corpo é lançado a partir do solo com velocidade inicial v=10m/s a um ângulo θ com a horizontal.

- *Para* θ de π/8 a 3π/8 com passo de π/80 :
  - *Plote (no mesmo gráfico) a posição (x,y) do corpo de t_1=0s com um passo de* ***Δt=0.1s*** *usando o método de Euler.*
- *Calcule o valor do alcance* $\boldsymbol{x_A}$ *para cada ângulo e faça um gráfico* $\boldsymbol{x_A}$ *vs* θ*.*
- *Qual o maior valor do alcance* $\boldsymbol{x_A}$ *encontrado? Para qual ângulo ocorre?*

Dica : Para cada $\boldsymbol{x_A}$ calculado, adicione-o a um vetor (concatenando):

vecxA=[vecxA xA]; % Adiciona xA ao final de vecxA